03-01-00

A

jc698 U.S. PTO 02/29/00

jc525 U.S. PTO 09/515700 02/29/00

IBM Docket No. FR9-99-001

In the United States Patent and Trademark Office
Patent Application Transmittal

Transmitted herewith for filing is the Patent Application of:

Inventors(s): Olivier Daude, Andrew Forth and Olivier Hericourt

For: Method and System for Optimally Selecting a Web Firewall in a TCP/IP Network

Enclosed are

28 pages of specification, including ***18*** claims, plus ***8*** sheets of ***formal*** drawings.

X An assignment of the invention to International Business Machines Corporation, Armonk, New York 10504.

X A certified copy of a/an ***European (serial no 99480011.8)*** application.

X Declaration and Power of Attorney.

X PTO-1449 & references

X A return post card

X Other: Preliminary Amendment, 9 pages

Filing Fee Calculation (For Other Than Small Entity)

Basic Fee:						$690.00
Claims Fees:	Filed	Limit	Extra		Rate per Extra	
Total claims:	18	20	0		$18.00	$0.00
Independent claims:	2	3	0		$78.00	$0.00
1	Multiple Dependent Claim Presented				$260.00	$260.00
					Total	**$950.00**

Please charge Deposit Account **09-0461** for the **Total** set forth above. The Commissioner is authorized to charge payment of any additional filing fees required under 37 CFR §1.16 and any patent application processing fees under 37 CFR §1.17 or to credit any overpayment to the identified account. A duplicate copy of this sheet is enclosed.

Express Mail Certificate

Express Mail Label No: EJ922405642US

Date: Feb. 29, 2000

I hereby certify that I am depositing the papers identified above with the U.S. Postal Service "Express Mail Post Office to Address" service on the above date, addressed to the Commissioner of Patents and Trademarks, Washington, DC 20231

Dianne Lane

Dianne Lane

BY: Jeanine S. Ray-Yarletts

Jeanine S. Ray-Yarletts

Attorney of Record Reg. No. 39,808

Date: Feb. 29, 2000

IBM Corporation T81/062
Intellectual Property Law
PO Box 12195
Res. Tri. Park, NC 27709

Telephone: 919-543-2541 FAX 919-254-4330

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of:	:	Date: February 29, 2000
O. Doude, et al.	:	IBM Corporation
Serial No.:	:	Dept. T81/Bldg. 062
Filed: herewith	:	P.O. Box 12195
Title: Method and System for	:	Research Triangle Park, NC 27709
Optimally Selecting a Web	:	
Firewall in a TCP/IP Network	:	

PRELIMINARY AMENDMENT

Honorable Commissioner of Patents and Trademarks
Washington, D.C. 20231

Sir:

Prior to the initial review, please amend the enclosed application for patent as follows:

In the Specification:

Page 1, line 15: please replace "language" with --protocol--.

Page 2, line 6: Please change "use" to --user--, insert --the-- after "integrates" and replace "format" with --formatting--.

Page 2, line 14: Please change "sound" to --audio-- and change "graphics" to --image (graphics)--.

Page 3, line 18: Please insert --servers-- after "Web".

Page 4, line 1: Please change “First” to --The first--.

Page 4, line 5: Please change “new version 1.1" to --newer versions (1.1)--.

Page 4, line 6: Please change “break” to --breaks--.

Page 4, line 11: Please insert --as-- between “such” and “a”.

Page 6, line 17: Please insert --a-- before “Socks Server”; insert --a-- before “Socks Gateway” and delete the “a” at the end of the line.

Page 6, line 26: Please change the first “a” to --as--.

Page 6, line 27: Please insert --a-- between “as” and “Proxy”.

Page 6, line 28: Please delete the “,”.

Page 6, line 31: Please insert --that-- between “proposes” and “the”; delete “to”.

Page 6, line 32: Please insert --of-- after “options”.

Page 7, line 10: Please change “refers” to --refer--.

Page 7, line 11: Please change “Proxies” to --Proxy--.

Page 8.line 33: Please change “has” to --having--.

Page 9, line 1: Please insert --the-- between “in” and “Javascript”.

Page 9, line 12: Please delete “incorporated herewith by reference”.

Page 10, line 22: Please delete "incorporated herewith by reference".

Page 11, line 10: Please change "Servers" to --Server--.

Page 12, line 24: Please change "performances are" to --performance is--.

Page 15, line 2: Please insert --the-- after "where".

Page 15, line 8: Please change "the" (second occupance) to --The--.

Page 16, line 14: Please change "measurements" to --measurement--.

Page 17, line 11: Please insert --a-- before "normal".

Page 19, line 5: Please insert --a-- before "Web".

Page 19, line 25: Please change "performances are" to --performance is--.

Page 28 (abstract): Please delete the paragraph indication between lines 12 and 13. Please also delete the reference to Figure 6.

In the Claims:

Claim 1 (Once amended) A method for dynamically selecting a firewall server [(603)] for a web client [(601)], in particular a web browser [(601)], in a Transmission Control Protocol/Internet Protocol (TCP/IP) network comprising the a plurality of firewall servers [(503)], said method comprising the steps of:

measuring performance and availability of each firewall server [(603)] using measurement probes [(607)]; and,

dynamically selecting a firewall server according to the performance and availability measurements [(607)].

Claim 2 (Once amended). The method according to [the preceding claim] claim 1 wherein the step of measuring the performance and availability of each firewall server [(603)] using measurement probes [(607)] comprises the further step of [: .] measuring the response time needed for retrieving from a web server [(605)] known information, in particular one or a plurality of known web pages, through each firewall server [(603);].

Claim 3 (Once amended). The method according to [the preceding claim] claim 2 wherein the step of measuring the response time comprises the further steps of:

establishing [(402)] a connection with the web server [(605)] through each firewall server [(603)];

retrieving [(403)] the one or a plurality of known web pages from the web server [(605)]; and,

checking [(405)] that the retrieved one or plurality of web pages contain one or a plurality of known keywords.

Claim 4 (once amended). The method according to [any one of the preceding claims] claim 1 or 3 wherein the step of measuring the performance of each firewall server [(603)] using measurement probes [(607)] comprises the further step of:

comparing each firewall server said measured response time with previous measured response times; and,

determining for each firewall [(603)] the degradation or the amelioration of the measured response time.

Claim 5 (once amended). The method according to [any one of the preceding claims] claim 1 or 3 wherein the step of measuring the availability of each firewall server using measurement probes [(607)] comprises the further step of:

detecting failures on each firewall server; and,

excluding firewall servers in failure from the step of selecting a firewall server.

Claim 6 (Once amended). The method according to [any one of the preceding claims] claim 1 or 3 wherein said firewall server [(603)] is a proxy server [(304)] or [/and] a socks server [(311)].

Claim 7 (Once amended). The method according to [any one of the preceding claims] claim 1 or 3 comprising the further steps of:

processing performance and availability measurements [(607)] from a single universal resource locator (URL) system [(606)]; and,

dynamically creating a configuration file based on the performance and availability measurements, preferably in the Javascript language, on said universal resource locator (URL) system [(606)] for selection said firewall server [(603)].

Claim 8 (Once amended). The method according to [any one of the preceding claims] claim 1 or 3 wherein the step of dynamically creating a configuration file is processed by a common gateway interface (CGI) [(608)] on said universal resource locator (URL) system [(606)].

Claim 9 (Once amended). The method according to [any one of the preceding claims] claims 1 or 3 wherein the step of selecting a firewall server [(603)] comprises the further step of [: .} downloading the configuration file from the universal resource

locator (URL) system [(606) to the web client, in particular to the] web browser [(601)].

Claim 10 (Once amended). The method according to [any one of the preceding claims] claim 1 or 3 wherein the steps of measuring performance and availability and of dynamically selecting a firewall server [(603)] are periodically processed in the universal resource locator (URL) system [(606)] and the configuration file created by the common gateway interface [(608)] (CGI) is periodically downloaded to the web client [(601)].

Claim 11 (Once amended). The method according to [any one of the preceding claims] claim 1 or 3 comprising the further steps of:

pre-selecting a backup firewall server [(603)] in a background process; and,

switching to said backup firewall server in case of failure of the selected firewall server.

Claim 12 (Once amended). The method according to [any one of the preceding claims] claim 1 or 3 wherein the step of selecting a firewall server according to performance and availability measurements comprises the further step of [: .] selecting the firewall server according to the Internet Protocol (IP) address.

Claim 13. Please delete this claim without prejudice.

Claim 14 (new).A program product for dynamically selecting a firewall server for a web client, in particular a web browser, in a Transmission Control Protocol/Internet Protocol (TCP/IP) network comprising the a plurality of firewall servers , said program product comprising the steps of:

programmatically measuring performance and availability of each firewall server using measurement probes; and,

dynamically, using programmatic means, selecting a firewall server according to the performance and availability measurements.

Claim 15 (new). The program product according to claim 1 wherein the step of measuring the performance and availability of each firewall server using measurement probes comprises the further step of programmatically measuring the response time needed for retrieving from a web server known information, in particular one or a plurality of known web pages, through each firewall server.

Claim 16 (new). The program product according to claim 15 wherein the step of measuring the response time comprises the further steps of:

programmatically establishing a connection with the web server through each firewall server;

programmatically retrieving the one or a plurality of known web pages from the web server; and,

programmatically checking that the retrieved one or plurality of web pages contain one or a plurality of known keywords.

Claim 17 (new). The program product according to claim 14 or 16 wherein the step of measuring the performance of each firewall server using measurement probes comprises the further step of:

programmatically comparing each firewall server said measured response time with previous measured response times; and,

programmatically determining for each firewall the degradation or the amelioration of the measured response time.

Claim 18(new). The program product according to claim 14 or 16 wherein the step of measuring the availability of each firewall server using measurement probes comprises the further step of:

programmatically detecting failures on each firewall server; and,

programmatically excluding firewall servers in failure from the step of selecting a firewall server.

Claim 19 (new). The program product according to claim 14 or 16 wherein said firewall server is a proxy server or a socks server.

REMARKS

The Specification has been amended to fix several grammatical and typographical errors that occurred in the priority filing. The claims have been amended to bring them into conformance with U.S. regulations and to present the associated system and program product claims disclosed in the invention. The abstract has been amended to conform with the requirement of being one paragraph of 50 words or less.

The applicants have made a sincere effort to place this application in condition for allowance. Early notification of such allowance is respectfully requested.

Respectfully submitted,

Jeanine S. Ray-Yarletts
Reg. No. 39,808
Attorney for the Applicants

:jsr

Phone: 919-543-2541

Fax: 919-254-4330

METHOD AND SYSTEM FOR OPTIMALLY SELECTING A WEB FIREWALL IN A TCP/IP NETWORK

Technical field of the invention

The present invention relates to computer networks, and more particularly to a method and system in a TCP/IP network for optimally selecting a Web Firewall according to some response time and availability criteria.

Background art

INTERNET

The Internet is a global network of computers and computers networks (the "Net"). The Internet connects computers that use a variety of different operating systems or languages, including UNIX, DOS, Windows, Macintosh, and others. To facilitate and allow the communication among these various systems and languages, the Internet uses a language referred to as TCP/IP ("Transmission Control Protocol/Internet Protocol"). TCP/IP protocol supports three basic applications on the Internet :

- transmitting and receiving electronic mail,
- logging into remote computers (the "Telnet"), and
- transferring files and programs from one computer to another ("FTP" or "File Transfer Protocol").

WORLD WIDE WEB

With the increasing size and complexity of the Internet, tools have been developed to help find information on the network, often called navigators or navigation systems. Navigation systems that have been developed include standards

such as Archie, Gopher and WAIS. The World Wide Web ("WWW" or "the Web") is a recent superior navigation system. The Web is:

- an Internet-based navigation system,
- an information distribution and management system for the Internet, and
- a dynamic format for communicating on the Web.

The Web seamlessly, for the use, integrates format of information, including still images, text, audio and video. A user on the Web using a graphical user interface ("GUI", pronounced "gooey") may transparently communicate with different host computers on the system, and different system applications (including FTP and Telnet), and different information formats for files and documents including, for example, text, sound and graphics.

HYPERMEDIA

The Web uses hypertext and hypermedia. Hypertext is a subset of hypermedia and refers to computer-based "documents" in which readers move from one place to another in a document, or to another document, in a non-linear manner. To do this, the Web uses a client-server architecture. The Web servers enable the user to access hypertext and hypermedia information through the Web and the user's computer. (The user's computer is referred to as a client computer of the Web Server computers.) The clients send requests to the Web Servers, which react, search and respond. The Web allows client application software to request and receive hypermedia documents (including formatted text, audio, video and graphics) with hypertext link capabilities to other hypermedia documents, from a Web file server.

The Web, then, can be viewed as a collection of document files residing on Web host computers that are interconnected by hyperlinks using networking protocols, forming a virtual "web" that spans the Internet.

UNIFORM RESOURCE LOCATORS

A resource of the Internet is unambiguously identified by a Uniform Resource Locator (URL), which is a pointer to a particular resource at a particular location. A URL specifies the protocol used to access a server (e.g. HTTP, FTP,..), the name of the server, and the location of a file on that server.

HYPER TEXT TRANSFER PROTOCOL

Each Web page that appears on client monitors of the Web may appear as a complex document that integrates, for example, text, images, sounds and animation. Each such page may also contain hyperlinks to other Web documents so that a user at a client computer using a mouse may click on icons and may activate hyperlink jumps to a new page (which is a graphical representation of another document file) on the same or a different Web server.

A Web server is a software program on a Web host computer that answers requests from Web clients, typically over the Internet. All Web use a language or protocol to communicate with Web clients which is called Hyper Text Transfer Protocol ("HTTP"). All types of data can be exchanged among Web servers and clients using this protocol, including Hyper Text Markup Language ("HTML"), graphics, sound and video. HTML describes the layout, contents and hyperlinks of the documents and pages. Web clients when browsing :

- convert user specified commands into HTTP GET requests,
- connect to the appropriate Web server to get information, and
- wait for a response. The response from the server can be the requested document or an error message.

After the document or an error message is returned, the connection between the Web client and the Web server is closed.

First version of HTTP is a stateless protocol. That is with HTTP, there is no continuous connection between each client and each server. The Web client using HTTP receives a response as HTML data or other data. This description applies to version 1.0 of HTTP protocol, while the new version 1.1 break this barrier of stateless protocol by keeping the connection between the server and client alive under certain conditions.

BROWSER

After receipt, the Web client formats and presents the data or activates an ancillary application such a sound player to present the data. To do this, the server or the client determines the various types of data received. The Web Client is also referred to as the Web Browser, since it in fact browses documents retrieved from the Web Server.

DOMAIN NAMES

The host or computers names (like www.entreprise.com) are translated into numeric Internet addresses (like 194.56.78.3), and vice versa, by using a method called DNS ("Domain Name Service"). DNS is supported by network-resident servers, also known as domain name servers or DNS servers.

INTRANET

Some companies use the same mechanism as the Web to communicate inside their own corporation. In this case, this mechanism is called an "Intranet". These companies use the same networking/transport protocols and locally based Web servers to provide access to vast amount of corporate information in a cohesive fashion. As this data may be private to the corporation, and because the members of the company still need to have access to public Web information, to avoid that people not belonging to the company can access to this private Intranet coming from the public Internet, they protect

the access to their network by using a special equipment called a Firewall.

FIREWALL

A Firewall protects one or more computers with Internet connections from access by external computers connected to the Internet. A Firewall is a network configuration, usually created by hardware and software, that forms a boundary between networked computers within the Firewall from those outside the Firewall. The computers within the Firewall form a secure sub-network with internal access capabilities and shared resources not available from the outside computers.

Often, a single machine, on which the Firewall is, allows access to both internal and external computers. Since the computer, on which the Firewall is, directly interacts with the Internet, strict security measures against unwanted access from external computers are required.

A Firewall is commonly used to protect information such as electronic mail and data files within a physical building or organization site. A Firewall reduces the risk of intrusion by unauthorized people from the Internet, however, the same security measures can limit or require special software for those inside the Firewall who wish to access information on the outside. A Firewall can be configured using "Proxies" or "Socks" to designate access to information from each side of the Firewall.

PROXY SERVER

A HTTP Proxy is a special server that typically runs in conjunction with Firewall software and allows an access to the Internet from within a Firewall. The Proxy Server :

- waits for a request (for example a HTTP request) from inside the Firewall,
- forwards the request to the remote server outside the Firewall,

- reads the response, and
- sends the response back to the client.

A single computer can run multiple servers, each server connection identified with a port number. A Proxy Server, like an HTTP Server or a FTP Server, occupies a port. Typically, a connection uses standardized port numbers for each protocol (for example, HTTP = 80 and FTP = 21). That is why an end user has to select a specific port number for each defined Proxy Server. Web Browsers usually let the end user set the host name and port number of the Proxy Servers in a customizable panel. Protocols such as HTTP, FTP, Gopher, WAIS, and Security can usually have designated Proxies. Proxies are generally preferred over Socks for their ability to perform caching, high-level logging, and access control, because they provide a specific connection for each network service protocol.

SOCKS

Socks Server (also called Socks Gateway) is also a software that allows computers inside a Firewall to gain access to the Internet. Socks is usually installed on a server positioned either inside or on the Firewall. Computers within the Firewall access the Socks Server as clients to reach the Internet. Web Browsers usually let the end user set the host name and port number of the Socks hosts (servers) in a customizable panel. On some Operating Systems, the host is specified in a separate file (e.g. socks.conf file). As the Socks Server acts a a layer underneath the protocols (HTTP, FTP, ..), it cannot cache data (as Proxy does), because it doesn't decode the protocol to know what kind of data, it is transferring.

OPTIONS

The Web Browser often proposes the end user to select between the different options "No Proxies", "Manual Proxy

Configuration", or "Automatic Proxy Configuration" to designate the conduit between his computer and the Internet.

- Users with a direct connection to the Internet should use the default, which is "No Proxies".
- If the Intranet is protected by one or several Firewalls, the end user may :
  - select one of these Firewalls as the elected Proxy, by entering its host name into the "Manual Proxy Configuration", or
  - automatically refers to the enterprise policy in terms of Proxies attribution between locations, by pointing to a common configuration file in a remote server. This is done by choosing the "Automatic Proxy Configuration" and by providing the Web Browser with the unique address of the common configuration file ("Universal Resource Locator" or "URL") located in the remote server.

Today, most of the Web Browsers are configured to forward all requests, even requests for internal hosts, through the Socks Firewall. So when the end user wants to have access to an internal Web-based application, his request travels to the Firewall, and is then reflected back into the internal network. This sends internal traffic on a long path, puts extra load on the Firewall and on the network, and worst of all, slows down the response time the end user sees from the applications and Web pages he is trying to access. This is called "non flexible" Socks access (when everything goes via the Socks Server).

MANUAL PROXY CONFIGURATION

The Manual Proxy configuration in the Web Browser is simple to process, but its main drawback is that the Firewall (or Proxy) selection is then static. There is no dynamic criterion for the Firewall selection, such as selection of the

Firewall providing the best response time. Firewall failures require a manual reconfiguration of the navigation software to point to another active Firewall, since the manual configuration usually only allows the definition of one single Firewall per protocol with no possibility to pre-configure a backup Firewall. In addition to the manual proxy configuration in the Web Browser, external procedures can be used to provide some kind of robustness in the Firewall selection. They rely for instance on the use of multiple Firewalls having the same name defined as aliases in the Domain Name Server (DNS). But this technique based on alias definition still has drawbacks since for instance the DNS is not always contacted for name resolution by Web Clients caching locally the name resolution. Other techniques using external hardware equipment such as load and request dispatcher provide more robustness and load balancing, but still have drawbacks such as the need for additional and costly hardware.

AUTOMATIC PROXY CONFIGURATION

Automatic Proxy Configuration (or also referred to as "autoproxy") can set the location of the HTTP, FTP, and Gopher Proxy every time the Web Browser is started. An autoproxy retrieves a file of address ranges and instructs the Web Browser to either directly access internal IBM hosts or to go to the Socks Server to access hosts on the Internet.

Automatic Proxy Configuration is more desirable than simple Proxy Server Configuration in the Web Browser, because much more sophisticated rules can be implemented about the way Web pages are retrieved (directly or indirectly). Automatic Proxy Configuration is useful to users, because the Web Browser knows how to retrieve pages directly if the Proxy Server fails. Also Proxy requests can be directed to another or multiple Proxy Servers at the discretion of the system administrator, without the end user has to make any additional changes to his Web Browser configuration. In general, these Proxy configuration files (also called autoproxy code) are

usually written in Javascript language. Autoproxy facility can also contain a file of address ranges for instructing the Web Browser to either directly access internal hosts or to go to the Socks Server to access hosts on the Internet. The Socks Server protects the internal network from unwanted public access while permitting access of network members to the Internet. One of the drawbacks of this "autoproxy" mechanism is that there is no proactive Firewall failure detection nor response time consideration.

More explanations about the domain presented in the above sections can be found in the following publications incorporated herewith by reference:

- "Java Network Programming" by Elliotte Rusty Harold, published by O'Reilly, February 1997.
- "Internet in a nutshell" by Valerie Quercia, published by O'Reilly, October 1997.
- "Building Internet Firewalls" by Brent Chapman and Elizabeth Zwichky, published by O'Reilly, September 1995.

PROBLEM

The problem to solve is to provide an optimized Web access, with a dynamic Proxy or Socks Server selection to get the best response time, and a detection of failures in Proxy or Socks Server to prevent Web service disruption. The current solutions address this problem partially:

- Web Browsers can be manually configured with the target Proxy or Socks Server. The main drawbacks of this solution are the following :
  - There is no dynamic Proxy/Socks Server selection. A manual reconfiguration of the Web Browser upon Proxy/Socks Server failure is required.
  - Only a "manual" load balancing through the Web Browser static configuration is provided.

- Proxy/Socks Server names must be known and manually configured by end users.

- Web Browsers can be configured with their autoproxy feature, using a static list of target Proxy/Socks Servers downloaded from a dedicated autoproxy URL (Uniform Resource Locator) system. The main drawback of this solution is the following:

  - There is no response time consideration in the Proxy/Socks Server selection, nor efficient Proxy/Socks Server failure detection (i.e. Web Browser waits for time-out before switching to backup, even at initial autoproxy loading) in the Proxy/Socks Server selection.

An alternate to these current solutions is to cluster the Proxy/Socks Servers using an external dispatcher system acting as single logical access point. All Web Browsers are then manually configured with the name of that external dispatcher system (as the target Proxy/Socks Server) which then routes the traffic to a selected Proxy/Socks Server. An example of such a dispatcher is for example the IBM Interactive Network Dispatcher product. More information concerning this product can be found in IBM's publication entitled "Interactive Network Dispatcher V1.2 - User's Guide" GC31-8496-01 incorporated herewith by reference. Although a dispatcher oriented solution allows an efficient load balancing in most cases, its main drawback is that additional dedicated system or specific hardware is required, and that the external dispatcher name has to be manually configured by the end users in their Web Browsers.

Objects of the invention

- The object of the present invention is to optimize Proxy/Socks Server selection by using availability and response time criteria.
- It is a further object of the present invention to optimize the Web service performance by integrating a response time factor to the Proxy/Socks Server selection.
- It is another object of the present invention to minimize Web service interruption and thus to insure a better service availability by automatically detecting Proxy/Socks Servers failures.

Summary of the invention

The present invention relates to dynamic autoproxy configuration and more particularly to a method and system for optimizing selection of a Proxy/Socks Server according to some response time and availability criteria. The invention rests on a dynamic autoproxy mechanism using availability and response time probes.

The present invention also relies on probes retrieving well known HTML pages through each Proxy/Socks Server, measuring associated response time, detecting Proxy/Socks failures and degradation of response time.

The present invention also uses a CGI (Common Gateway Interface) program for dynamically creating autoproxy code (in a preferred embodiment Javascript code) on an autoproxy URL (Universal Resource Locator) system for selecting the Proxy/Socks Server using availability and response time information provided by probes.

The present invention fixes the drawbacks of the existing current solutions by integrating dynamic Proxy/Socks availability and response time selection criteria to the autoproxy mechanism.

The present inventions provides the following advantages:

- Early detection of Proxy/Socks Servers failures provides a high Web service availability.
- Integration of a response time factor to the Proxy/Socks Server selection optimizes the Web service performances.
- Induced HTTP survey traffic is minimized by running availability and response time probes from a single autoproxy URL system (compared with running the probes on each Web Browser system).
- Integration of response time degradation in the probes achieves a proactive Proxy/Socks Servers failure detection.
- Periodical dynamic update of "best" Proxy/Socks Server can be provided to Web Browser.
- Useless traffic to failing Proxy/Socks Server is minimized since Proxy/Socks Servers are excluded from list of available target servers upon failure detection.
- No additional or specific hardware is required.
- Ease of Web Browser configuration provided to mobile users (Web Browser is configured once).
- Web Browser performances are not degraded because availability and response time probes are not processed within the downloaded autoproxy code (Javascript code) but in the autoproxy URL system.

Drawings

The novel and inventive features believed characteristics of the invention are set forth in the appended claims. The

invention itself, however, as well as a preferred mode of use, further objects and advantages thereof, will best be understood by reference to the following detailed description of an illustrative detailed embodiment when read in conjunction with the accompanying drawings, wherein :

- Figure 1 is a general logical view of an end user system interfacing a Web Browser for accessing the World Wide Web according to prior art.

- Figure 2 is a general physical view of the set-up shown in Figure 1, according to prior art.

- Figure 3 is a logical view of the availability and response time probes external flows according to the present invention.

- Figure 4 is a flow chart showing the internal logic flow of the availability and response time probe introduced in Figure 3 according the present invention.

- Figure 5 is a physical view of the logical environment described in Figure 3 according to the present invention.

- Figure 6 is a view of the data flows associated with the entities depicted in Figure 5, according to the present invention.

- Figure 7 depicts the storage of the availability and response times probes measurements, according to the present invention.

- Figure 8 is a flow chart of the program running on the autoproxy URL (Universal Resource Locator) system, according to the present invention.

Preferred embodiment of the invention

The present invention relies on dynamic autoproxy configuration and more particularly to a method and system for selecting a Proxy/Socks Server according to some response time and availability criteria. It rests on a dynamic autoproxy mechanism using availability and response time probes. It relies on probes retrieving well known HTML pages through each Proxy/Socks Server, measuring associated response time, detecting Proxy/Socks failures and degradation of response time.

It also uses a CGI (Common Gateway Interface) program for dynamically creating autoproxy code (in a preferred embodiment Javascript code) on an autoproxy URL (Universal resource locator) system for selecting said Proxy/Socks Server.

LOGICAL VIEW OF A END USER ACCESSING THE WORLD WIDE WEB

Figure 1 shows a user system with a user interface (102) running a program known as a Web Browser (101) which enables access to the World-Wide-Web (WWW). The WWW content is transferred using the HTTP protocol. HTTP requests and responses are going to and from the Web Browser program (101) and a destination Web Server (103) containing the WWW content the user tries to access. The Firewall (104) between the Web Browser (101) and the Web Server (103) acts as an intermediary HTTP Proxy forwarding the HTTP requests and responses to their destination. The Web Browser program (101) makes an HTTP request to the Firewall (104) and the Firewall forwards the request to the destination Web Server (103). The flow in the reverse direction is the HTTP response which again goes via the Firewall (104) on its way to the Web Browser (101). In this way the Firewall can limit the traffic to the transactions it is configured to allow (based on some defined

security and access control policy). The Firewall hence protects the network where Web Browser is located.

GENERAL PHYSICAL VIEW OF AN END USER ACCESSING THE WWW

Figure 2 is a physical view of the set-up shown logically in Figure 1. In this particular example, the Web Browser (201) runs on a system attached to an Intranet (202). The Firewalls (203) that protect the Intranet attach both the (private) Intranet (202) and the (public) Internet (204). the destination Web Server (205) also connects to the Internet. This is the environment where the Web Browser, Firewalls, and Web Server perform their function when the user is "browsing" the Internet WWW. It is important to note the fact the Firewalls attach two networks and hence are able to act as the intermediary for communications between the two networks. Multiple Firewalls are often uses in order to provide some degree of access robustness and load sharing.

LOGICAL VIEW OF AVAILABILITY AND RESPONSE TIME PROBES

The domain of the invention is the one described in Figure 1 and Figure 2, where a user within an Intranet wants to access the World-Wide-Web using a Web Browser, and where the Intranet network is protected from the Internet by several Firewalls playing role of so called HTTP Proxies (Figure 2). The issue is to select the "best" Proxy/Socks Server to insure an optimized availability and response time of the service to the end user. To automatically optimize this selection, a software component called a "WWW availability and response time probe" is introduced. Its role is to provide selection criteria. As shown in Figure 3, this data is gathered by measuring the response time for requesting a specific content of a well known Web Server. The induced HTTP survey traffic is minimized, by running the availability and response time probes from single autoproxy URL system (versus running the probes on each Web Browser client system).

Figure 3 demonstrates the function of a flexible WWW availability and response time probe and the way it can be used to gather measurements on the availability and response time of both HTTP Proxies and Socks Servers. The upper part of Figure 3 details the interaction of the probe with an HTTP Proxy Server (304). The client system (302) that runs the probe (configured to test proxies) basically requests Web content (page) from the Web Server (307) via the Proxy Server (304) similar to the process shown in Figure 1. The HTTP request in this case represents an "HTTP survey flow" (303) to the Proxy Server. The Proxy Server forwards (306) the request to the Web Server (via the Firewall (305) which is not depicted). The client system times how long the request/response HTTP survey flow takes and uses this information as a measurement of the response time and availability via the tested Proxy Server (for the Web content samples that were tested). If the client system is also the autoproxy URL system (301) then this measurement information for each Proxy Server can be used to work out a sense of the "best" Proxy Server to use. This can then be encoded in the autoproxy URL that the Web Browser programs use to work out their correct Proxy Server to use.

The lower portion of the Figure 3 shows a similar arrangement but in this case the measurements data is being gathered for a Socks Server (Gateway) access method. Again a client probe (309) makes an HTTP request that represents an "HTTP survey flow" (310) which travels via the Socks Server (311) and then onto (312) the destination Web site (313). This HTTP request is for a set target URL (308) that is known to exist on the target Web Server. Again it is the timing of how long this survey takes that provides the measurements data that can be used to generate an autoproxy URL that takes into account the relative performance of a set of Socks Server (or in the case above, HTTP Proxies).

Obviously if there is no response to the HTTP survey flow, then the particular Proxy or Socks Server being tested

can be marked as unavailable. In this way the autoproxy URL can be used to not select Proxy or Socks Servers that do not work .

INTERNAL LOGIC OF THE AVAILABILITY AND RESPONSE TIME PROBES

The internal mechanism of the probe itself is described in Figure 4. The probe simulates a Web Client, by requesting through an HTTP connection a Web page from a target URL through the target Proxy/Socks Server (using its host name and port as a reference). The Web page is retrieved either through a normal HTTP connection, or through a socksified flow (a flow through a Socks Server). Typically, normal flow is used to retrieve a Web page from a Proxy Server or from a Web Server, while socksified flow is used to retrieve a Web page through a Socks Server. Then, the probe basically checks that the Web page :

- is received within an allowed amount of time in seconds, and
- contains a specific keyword to make sure that the received page is correct.

When these two conditions are fulfilled, the Web page retrieval is successful.

Finally, the probe returns either the associated response time in seconds (successful retrieval) or a failure return code. This mechanism retrieves one or multiple target Web pages. When multiple Web pages are retrieved, the probe program sequentially tests each Web page until one Web page retrieval is successful or all Web page retrievals fail. Probes :

- retrieve well known HTML pages through each Proxy/Socks Server,
- measure associated response time, and also

- detect Proxy/Socks Server failures and response time degradation.

Figure 4 is a flow chart showing the internal logic flow of the WWW availability and response time probe introduced in Figure 3.

- The first thing the probe program does is to start a timer (401).
- Next the probe program attempts to establish a connection (402) with the target Web Server to retrieve a Web page at the target URL (Universal Resource locator). The probe program establishes the connection according to the way it has been configured e.g. via an HTTP Proxy Server, via a Socks Server (Gateway) or directly.
- If the attempt for establishing the connection is unsuccessful, the probe program immediately goes into error mode (408). An error value is returned (407) by the probe program indicating that the connection is not possible.
- If the attempt for establishing the connection is successful, then the Web page (403) is retrieved by the probe program.
- The probe program then closes the connection (404) pursuant to the normal HTTP protocol procedure.
- To ensure that the Web page has been correctly retrieved, the probe program then searches for known keywords (405) that are expected to be in the Web page.
- If the keyword is found (406) in the Web page, then the Web page retrieval is successful. The timer is stopped and the correct response time for the operation is returned. By storing and integrating a short historic of the measured response time over time, the probe program can detect and return any response time degradation, thus enabling an anticipation of the Proxy/Socks Servers failures.

- If however the correct keyword is not found (407) in the Web page, then the Web retrieval is unsuccessful and again an error value is returned. The type of event that might trigger this sort of error is when the connection is successfully established but Web page with an error is retrieved.
- The action whereby the probe goes into retry mode (409) occurs only when the probe is configured to try multiple destination URL's as opposed to a single URL. This adds some robustness to the testing of the probe and hence insulates it somewhat from one-off network "glitches" (e.g. dropped connections etc.).

PHYSICAL VIEW OF AVAILABILITY AND RESPONSE TIME PROBES EXTERNAL FLOWS

The probes are used by various components and in various flows (Figures 5 and 6) in order to provide the Web Browser with the best Proxy/Socks Server. The data gathered by the probes are indirectly downloaded to the Web Browser by using an autoproxy mechanism. The present invention allows a software implementation with no additional or specific hardware.

The output from the probe is stored on the autoproxy URL system as shown in Figure 7 and used to create the autoproxy code (Javascript code in a preferred embodiment). There is no extra process inside the code. Web Browser performances are not degraded because availability and response time probes are not processed within the downloaded autoproxy code (Javascript code) but in the autoproxy URL system.

A CGI (Common Gateway Interface) program dynamically creates the autoproxy code as shown in Figure 8 with the availability and response time information provided by probes. The use of response time and availability criteria for selecting a Proxy/Socks Server by the probes is fully compatible, and can be combined, with existing criteria such as client's origin IP subnet.

The use of response time and availability criteria also provides a proactive Proxy/Socks Servers failure detection through the integration of response time degradation. The Web Browser can be periodically and dynamically updated with a new selection of the "best" Proxy/Socks Server using :

- "refresh" tag in the autoproxy code,
- external code (or Java applet), or
- a new feature in the Web Browser for periodically and automatically refreshing the autoproxy code.

Another positive consequence is the minimization of the useless traffic to failing Proxy/Socks Server since Proxy/Socks Servers are excluded from list of available target servers upon failure detection. Since an autoproxy mechanism is used, there is no need for manually updating the manual proxy configuration in the Web Browser in case of Proxy/Socks Server failure. Proxy/Socks Servers names or locations don't need to be known and configured by the end user, thus providing for instance a seamless service for mobile users.

Figure 5 is a physical view of the logical environment described in Figure 3. The Web Browser (501) attached to the Intranet (502) is configured to use an autoproxy URL to determine which Proxy/Socks Server (Firewall) (503) to use for having access to the Internet (504) and the destination Web Server (505). The system where the autoproxy URL resides (506) also runs the availability and response time probes (507) configured to test the Proxy/Socks Servers. The autoproxy URL uses the CGI (Common Gateway Interface) (508) to dynamically generate the autoproxy code of the autoproxy URL. The autoproxy code is based on the information gathered by the availability and response time probes. In this way the Web Browser is configured with

- an available Proxy/Socks Server, and

- what is deemed the best Proxy/Socks Server.

DATA FLOWS OF AVAILABILITY AND RESPONSE TIME PROBES

Figure 6 is a view of the actual data flows associated with the entities depicted in Figure 5. Again, the Web Browser (601) attached to the Intranet (602) is configured to use an autoproxy URL to determine which Proxy/Socks Server (Firewall) (603) to use for having access to the Internet (604) and the destination Web Server (605). The Web Browser has access (609) to the autoproxy URL system (606) to first determine which Proxy/Socks Serverl it should use. The Web Browser can be periodically and dynamically updated with the "best" Proxy/Socks Server using :

- the "refresh" tag in the autoproxy code,
- an external code (or Java applet), or
- a Web Browser new feature for periodically and automatically refreshing the autoproxy code.

The system where the autoproxy URL resides and which runs the availability and response time probes (607), uses the CGI (Common Gateway Interface)(608) to dynamically generate the autoproxy code (610) of the autoproxy URL. The autoproxy code is based on the information gathered by the availability and response time probes that have tested the Proxy/Socks Servers via the HTTP survey flows (611 and 612) described in Figure 3. In this way the Web Browser ends up with what is deemed the best Proxy/Socks Server.

INTERNAL STORAGE OF AVAILABILITY AND RESPONSE TIME PROBES

Figure 7 depicts the internal storage within the autoproxy URL system of the information retrieved by the availability and response times probes (701). Each probe updates (702) a table in the autoproxy URL system (703) with the measurements of each Proxy/Socks Server it tests. In this way the table contains the current state of all Proxy/Socks

Servers that are candidate to be selected and used by the Web Browser. At configurable or periodical time intervals, probes test again the Proxy/Socks Servers (704) and the cycle again is repeated.

PROGRAM RUNNING AT AUTOPROXY URL SYSTEM

Figure 8 again refers to the internal logic of the program running on the autoproxy URL system.

- The autoproxy URL system is initially contacted by a Web Browser (801) wanting to "know" which is the best Proxy/Socks Server (or Firewall) to use. This is for instance achieved by selecting the Automatic Proxy Configuration option in the Web Browser and by providing information such as the URL of the autoproxy code.
- The autoproxy URL system activates (802) the CGI (Common Gateway Interface) program (via the Web Server CGI extensions). The CGI program has access to all standard CGI variables including the IP (Internet Protocol) address of the Web Browser.
- The CGI program selects (803) the best Proxy/Socks Server for the client system (Web Browser) based on both the IP address of the Web Browser (obtained as a CGI variable) and the information generated by the availability and response time probes for each Proxy/Socks Server and stored in the table of the autoproxy URL system (807). The IP address is used to add a geographical criteria to the Proxy/Socks Server selection. For instance, if two Proxy/Socks Servers provide the same response time (one in the US, the other one in Europe), the closest Proxy/Socks Server is preferred (the one in Europe if the Web Browser is in Europe).
- To improve the robustness of the Proxy/Socks Server selection, the CGI program selects a best "backup" Proxy/Socks Server for the Web Browser (804). This "backup" Proxy/Socks Server is automatically used by the Web Browser after it times out attempting to use what it thinks is the

"best" Proxy/Socks Server. Again this "backup" Proxy/Socks Server is selected using both the IP address of the Web Browser (obtained as a CGI variable) and the information generated by the availability and response time probes for each Proxy/Socks Server and stored in the table of the autoproxy URL system (807).

- Once the CGI program has selected the best and backup Proxy/Socks Servers, it created the autoproxy code (805). This code is generally made of Javascript language.
- Once the autoproxy code has been created, the autoproxy URL system downloads it to the Web Browser (806) via standard HTTP protocol as any other output of a CGI program.

While the invention has been particularly shown and described with reference to a preferred embodiment, it will be understood that various changes in form and detail may be made therein without departing from the spirit, and scope of the invention.

Claims

1. A method for dynamically selecting a firewall server (603) for a web client (601), in particular a web browser (601), in a Transmission Control Protocol/Internet Protocol (TCP/IP) network comprising a plurality of firewall servers (503),
said method comprising the steps of :

   - measuring performance and availability of each firewall server (603) using measurement probes (607);

   - dynamically selecting a firewall server according to the performance and availability measurements (607).

2. The method according to the preceding claim wherein the step of measuring the performance and availability of each firewall server (603) using measurement probes (607) comprises the further step of:

   - measuring the response time needed for retrieving from a web server (605) known information, in particular one or a plurality of known web pages, through each firewall server (603);

3. The method according to the preceding claim wherein the step of measuring the
response time comprises the further steps of:

   - establishing (402) a connection with the web server (605) through each firewall server (603);

- retrieving (403) the one or a plurality of known web pages from the web server (605);

- checking (405) that the retrieved one or plurality of web pages contain one or a plurality of known keywords.

4. The method according to any one of the preceding claims wherein the step of measuring the performance of each firewall server (603) using measurement probes (607) comprises the further step of:

    - comparing for each firewall server said measured response time with previous measured response times;

    - determining for each firewall (603) the degradation or the amelioration of the measured response time.

5. The method according to any one of the preceding claims wherein the step of measuring the availability of each firewall server using measurement probes (607) comprises the further step of:

    - detecting failures on each firewall server;

    - excluding firewall servers in failure from the step of selecting a firewall server.

6. The method according to any one of the preceding claims wherein said firewall server (603) is a proxy server (304) or/and a socks server (311).

7. The method according to any one of the preceding claims comprising the further steps of:

- processing performance and availability measurements (607) from a single universal resource locator (URL) system (606);

- dynamically creating a configuration file based on the performance and availability measurements, preferably in Javascript language, on said universal resource locator (URL) system (606) for selecting said firewall server (603).

8. The method according to any one of the preceding claims wherein the step of dynamically creating a configuration file is processed by a common gateway interface (CGI) (608) on said universal resource locator (URL) system (606).

9. The method according to any one of the preceding claims wherein the step
of selecting a firewall server (603) comprises the further step of:

- downloading the configuration file from the universal resource locator (URL) system (606) to the web client, in particular to the web browser (601).

10. The method according to any one of the preceding claims wherein the steps of measuring performance and availability and of dynamically selecting a firewall server (603) are periodically processed in the universal resource locator (URL) system (606) and the configuration

file created by the common gateway interface (608)(CGI) is periodically downloaded to the web client (601).

11. The method according to any one of the preceding claims comprising the further steps of:

- pre-selecting a backup firewall server (603) in a background process;
- switching to said backup firewall server in case of failure of the selected firewall server.

12. The method according to any one of the preceding claims wherein step of selecting a firewall server according to performance and availability measurements comprises the further step of:

- selecting the firewall server according to the Internet Protocol (IP) address.

13. A system comprising means adapted for carrying out the method according to any one of the preceding claims.

METHOD AND SYSTEM FOR OPTIMALLY SELECTING A WEB FIREWALL IN A TCP/IP NETWORK

Abstract

The present invention relies on dynamic autoproxy configuration and more particularly to a method and system for selecting a Proxy/Socks Server according to some response time and availability criteria. It rests on a dynamic autoproxy mechanism using availability and response time probes. It relies on probes retrieving well known HTML pages through each Proxy/Socks Server, measuring associated response time, detecting Proxy/Socks failures and degradation of response time.

It also uses a CGI (Common Gateway Interface) program for dynamically creating autoproxy code (in a preferred embodiment Javascript code) on an autoproxy URL (Universal resource locator) system for selecting said Proxy/Socks Server.

Figure 6

Logical view of a end user accessing the World-Wide-Web

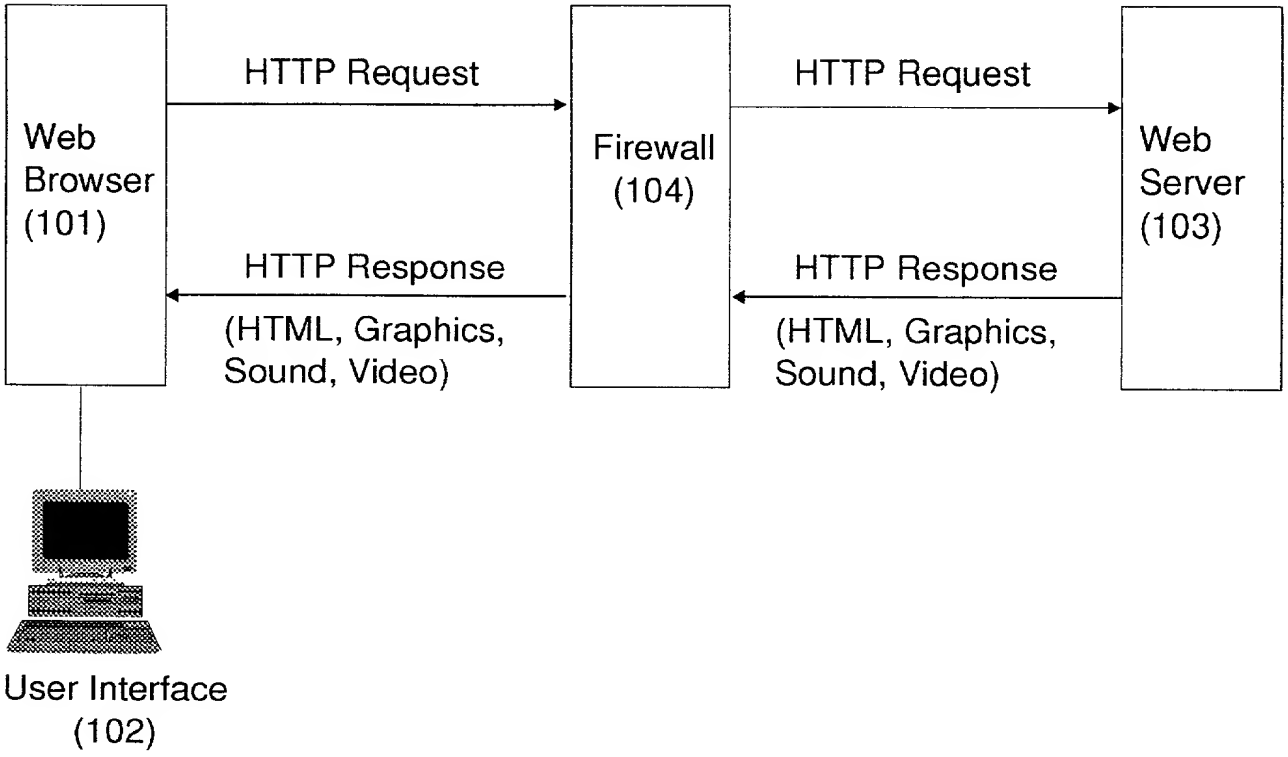

FIG. 1

General physical view of a end user accessing the World-Wide-Web

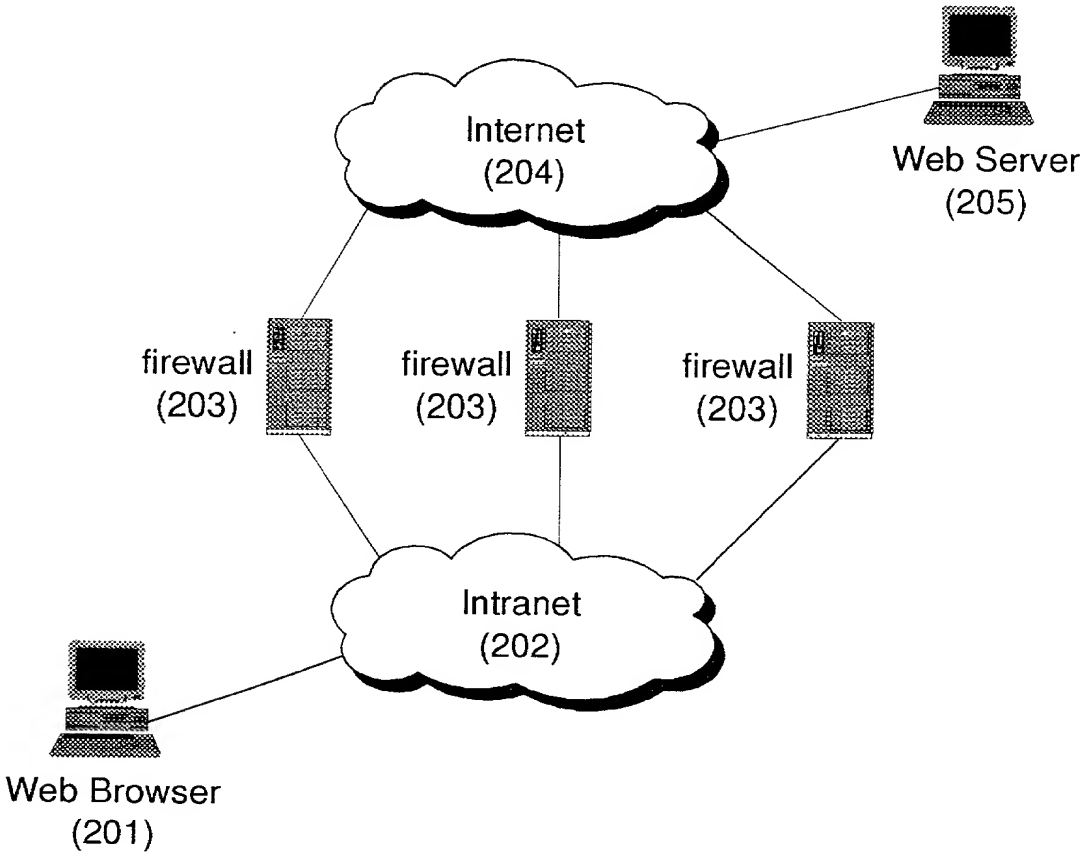

FIG. 2

Logical view of availability and response time probes external flows

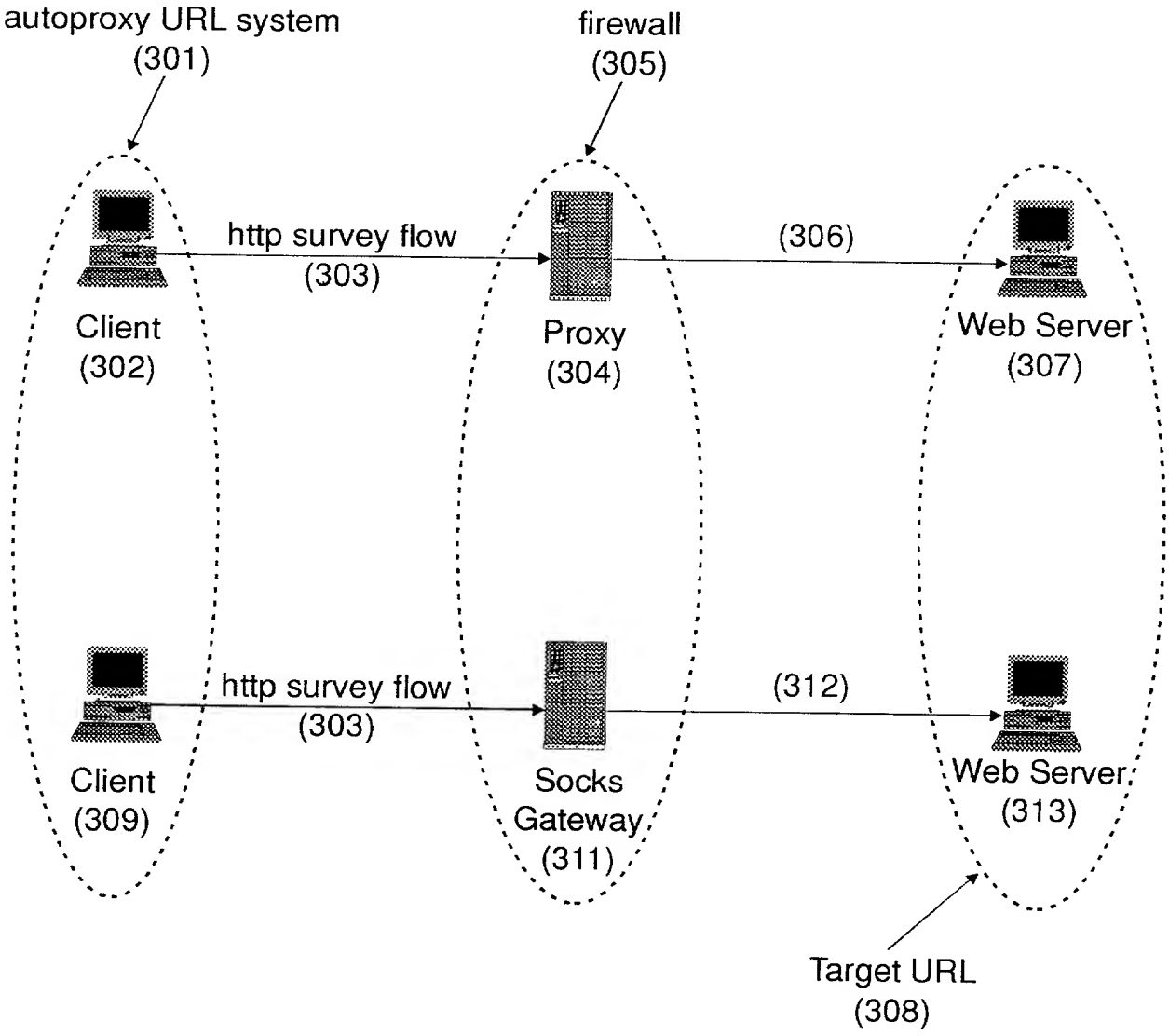

FIG. 3

Flow chart of internal logic of availability and response time probe

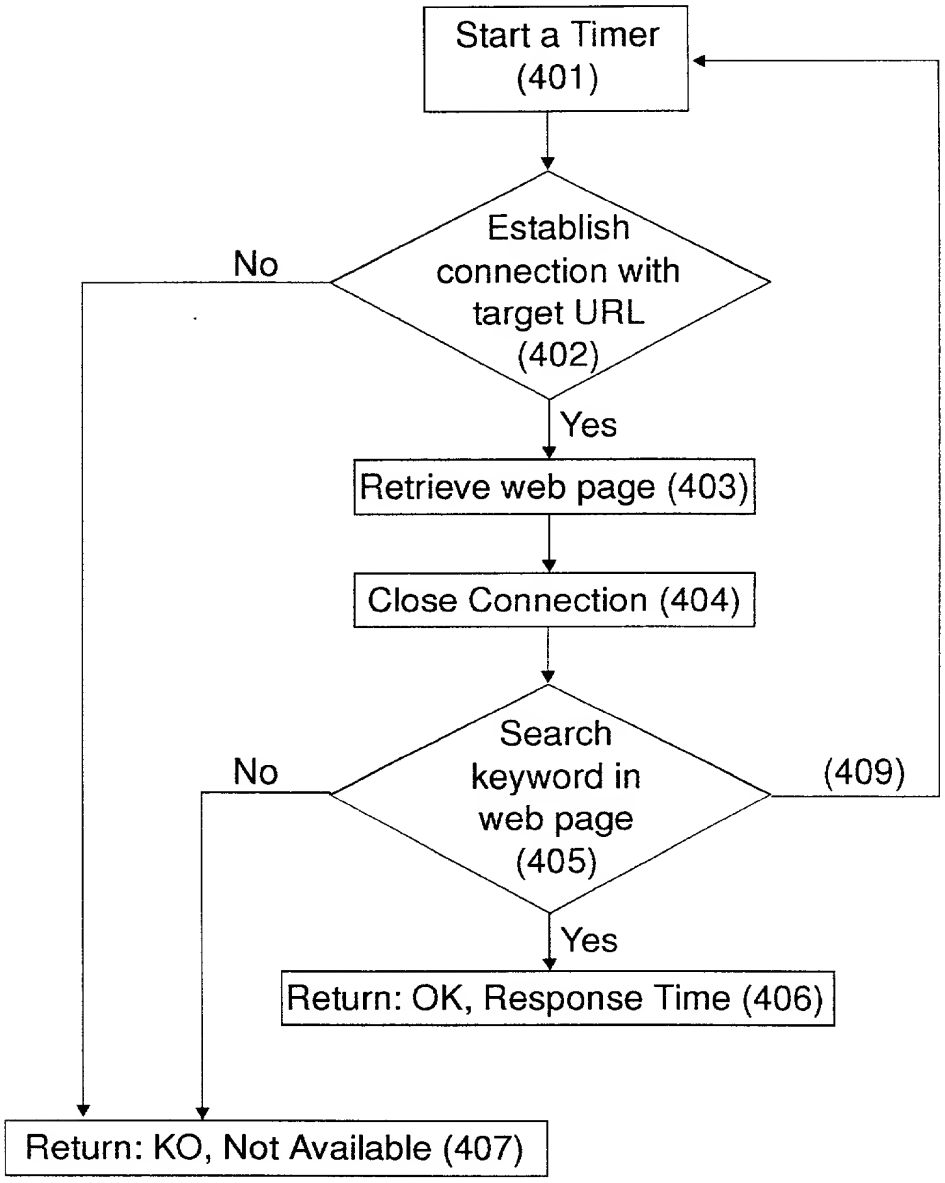

FIG. 4

Physical view of availability and response time probes external flows

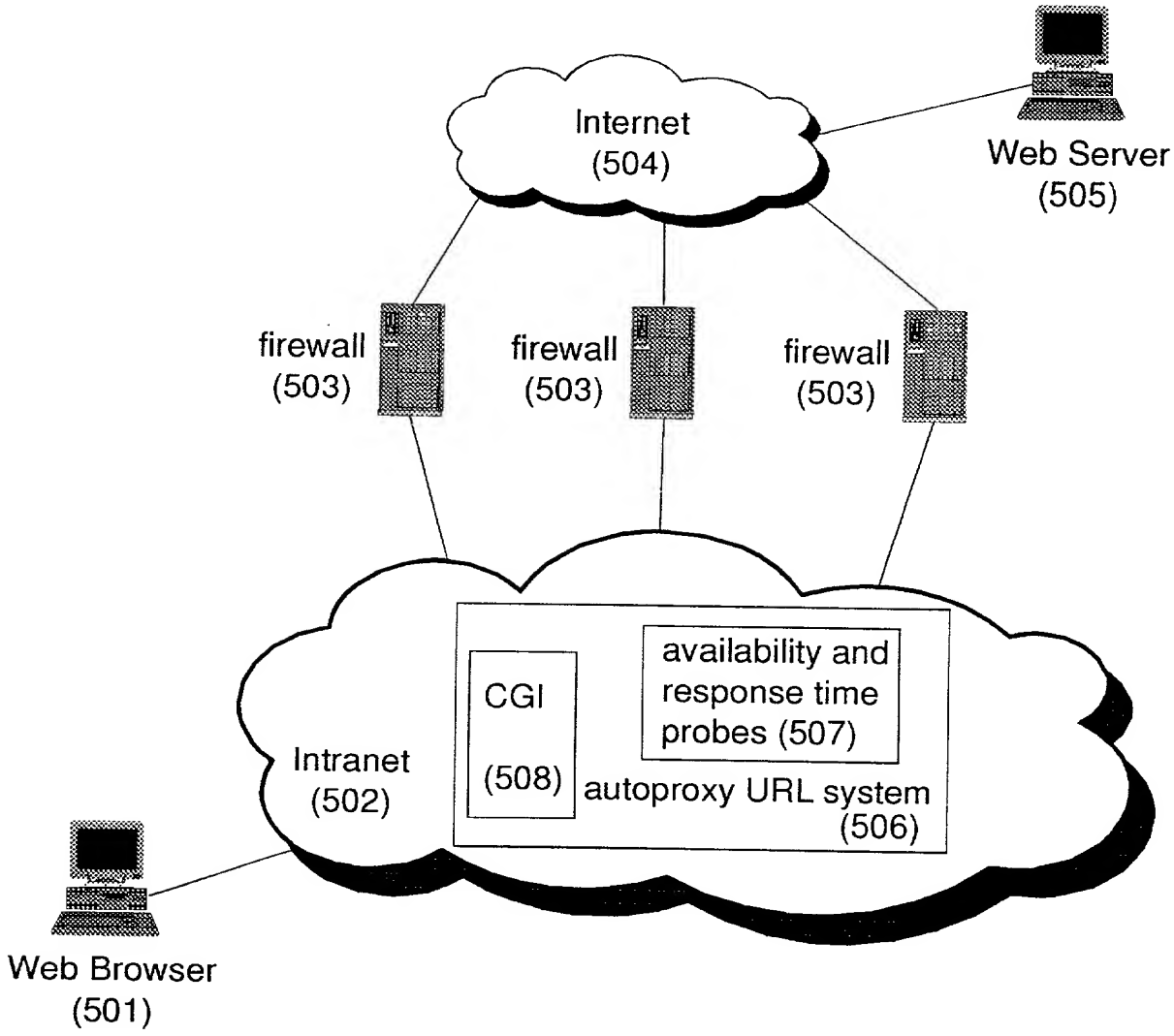

FIG. 5

Data flows of availability and response time probe

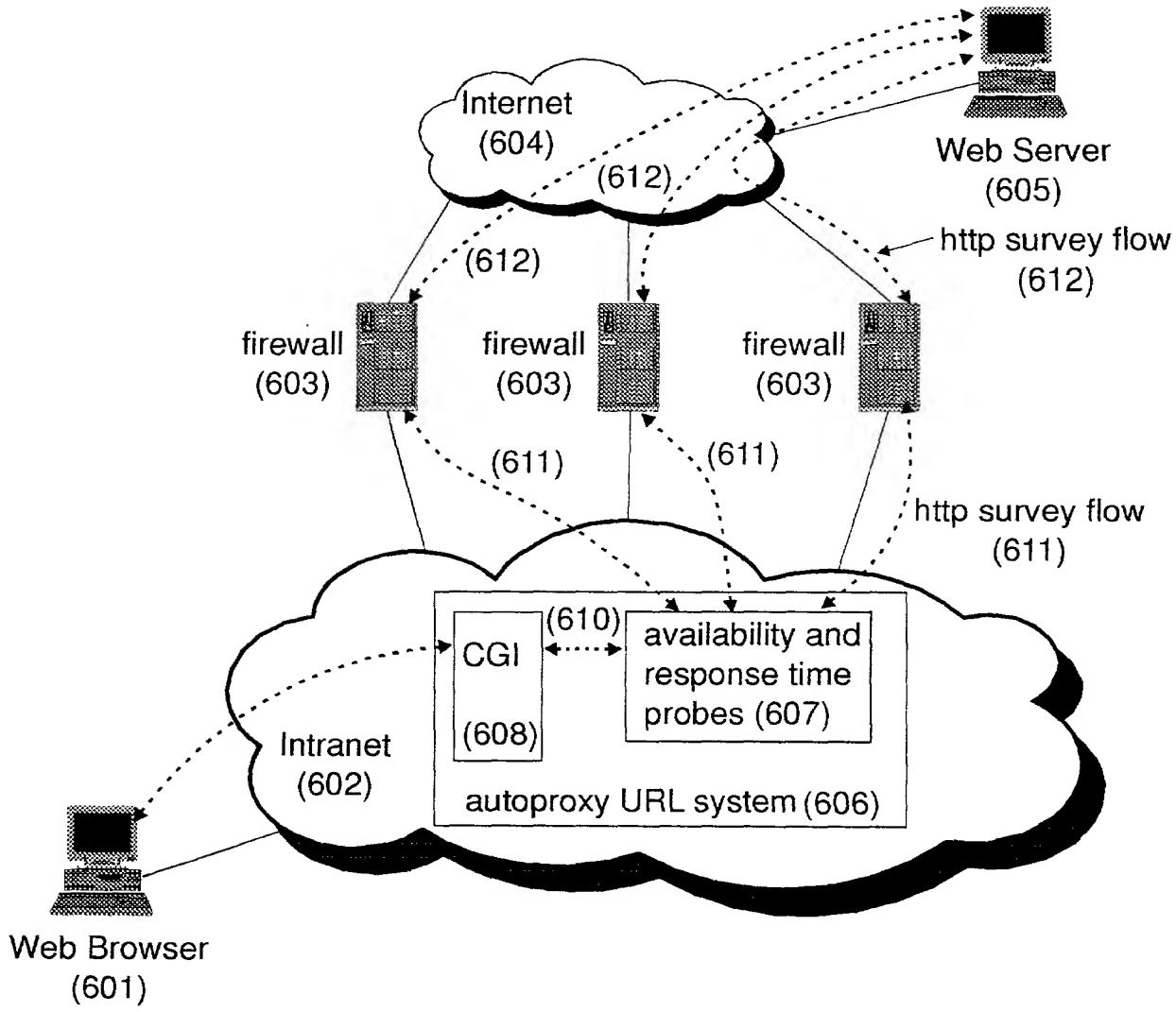

FIG. 6

Internal storage of WWW availability and response time probes

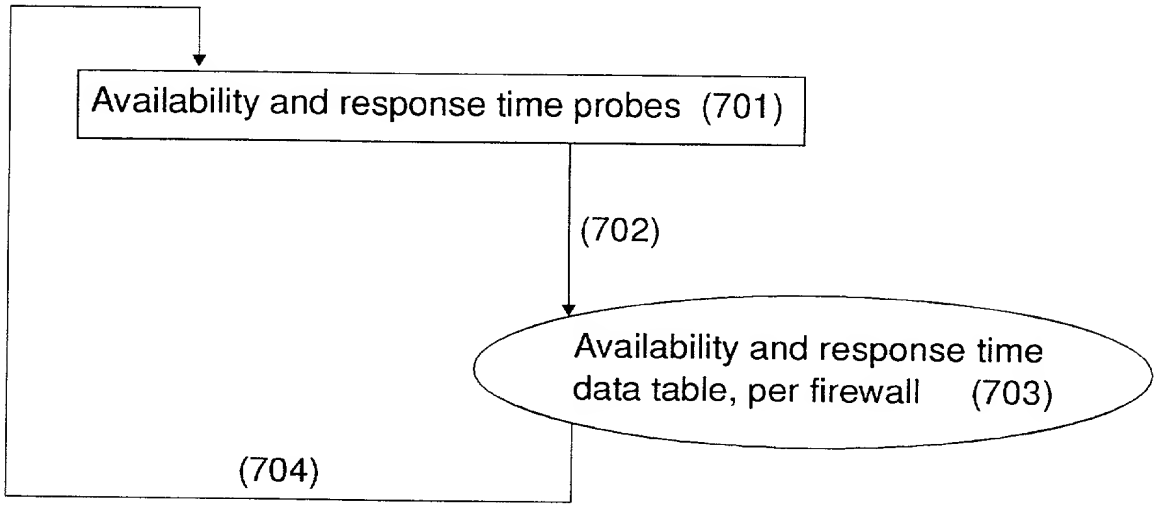

FIG. 7

Flow chart of the program running at autoproxy URL system

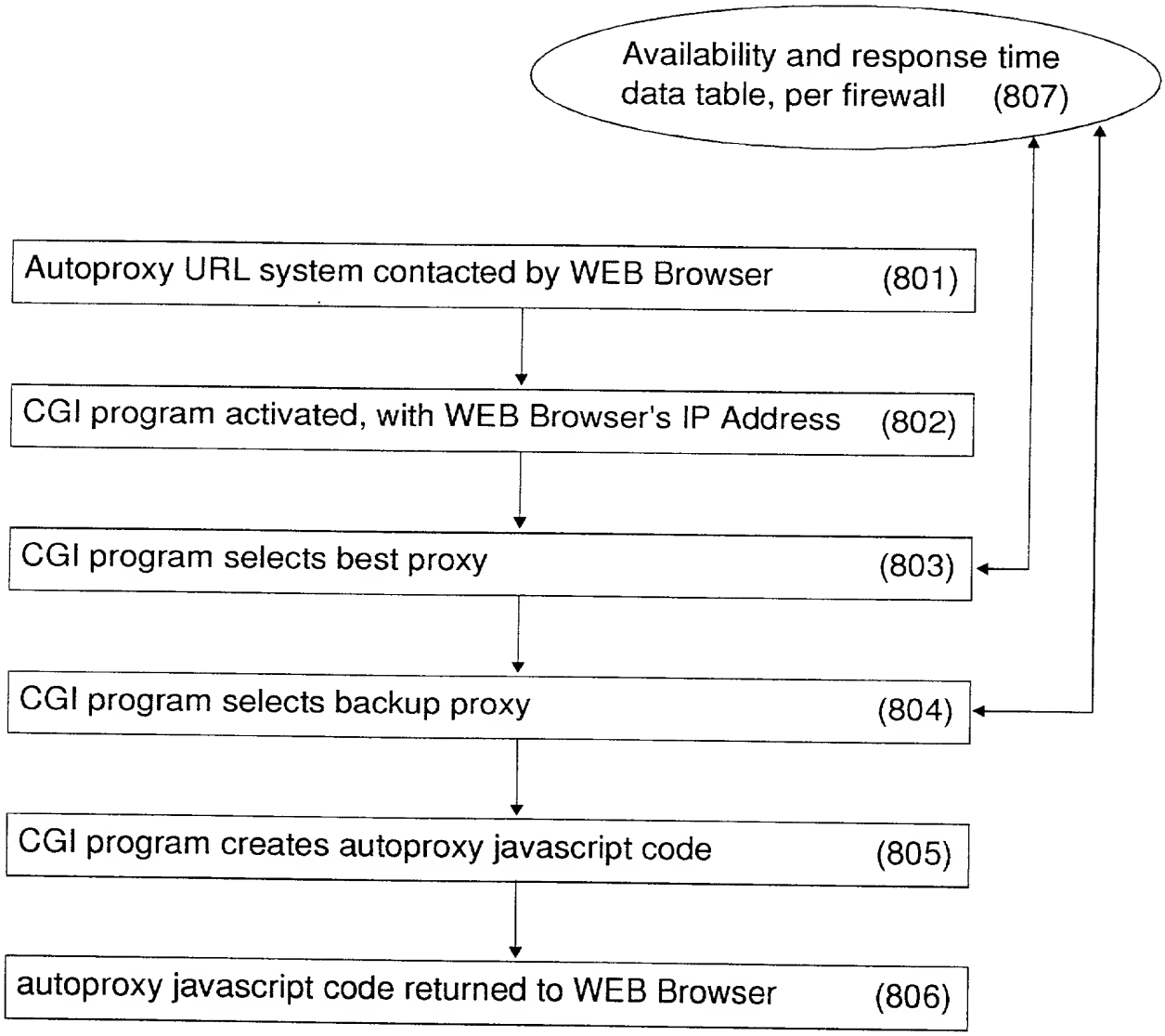

FIG. 8

Declaration and Power of Attorney for Patent Application

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name; I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

Method and System for Optimally Selecting a Web Firewall in a TCP/IP Network

the specification of which (check one)

[X] is attached hereto.

[ ] was filed on _____ as Application Serial No. _____ and was amended on _____.

I hereby state that I have reviewed and understand the contents of the above- identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the patentability of this application in accordance with Title 37, Code of Federal Regulations, §1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, §119 of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s):

Number	Country	Day/Month/Year	Priority Claimed
98480011.8	Europe	05/Mar/1999	Yes

I hereby claim the benefit under Title 35, United States Code, §120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, §112, I acknowledge the duty to disclose information material to the patentability of this application as defined in Title 37, Code of Federal Regulations, §1.56 which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

Prior U.S. Applications:

Serial No.	Filing Date	Status

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both,

under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

As a named inventor, I hereby appoint the following attorneys and/or agents to prosecute this application and transact all business in the Patent and Trademark Office connected therewith: J. S. Ray-Yarletts, Reg. No. 39,808; B. A. Clay, Reg. No. 32,121; G. M. Doudnikoff, Reg. No. 32,847; E. H. Duffield, Reg. No. 25,970; J. W. Herndon, Reg. No. 27,901; G. W. Woods, Reg. No. 24,144; C. A. Hughes, Reg. No. 26,914; E. A. Pennington, Reg. No. 32,588; J. E. Hoel, Reg. No. 26,279; and J. C. Redmond, Jr., Reg. No. 18,753.

Send all correspondence to: Jeanine S. Ray-Yarletts
IBM Corp., Dept. T81/Bldg. 062
P.O. Box 12195
Research Triangle Park, NC 27709
Phone: 919-543-2541 Fax: 919-254-4330

---

(1) Inventor: **Olivier Daude**

Signature: Olivier Daude

Date 01/06/2000

Residence: Residence La Vigie
169, avenue de Fabron
06200 Nice - France

Citizenship: France

Post Office Address: same as residence

(2) Inventor: **Andrew Forth**

Signature: 2ND FEB 2000

Date

Residence: 113, Cedar Terrace
Richmond
Surrey - TW9 2BY - United Kingdom

Citizenship: England

Post Office Address: same as residence

(3) Inventor: **Olivier Hericourt**

Signature: 01/05/2000 Hericourt
Date

Residence: Residence Le Flore
62, chemin du Val Fleuri
06800 Cagnes sur Mer - France

Citizenship: France

Post Office Address: same as residence